A PROOF OF LEMMA 3.1

Lemma 4.1. We first assume that the learnt functions $\psi_r^k : \mathbb{R}^{2d} \to \mathbb{R}^d, \psi_a^k : \mathbb{R}^d \to \mathbb{R}^d$ have bounded gradients. In other words, there exists $A, R > 0$, such that the following Jacobian matrices have bounded matrix norm:

$$J_{\psi_r^k}(\boldsymbol{x}, \boldsymbol{y}) = \begin{pmatrix} \frac{\partial \psi_{r,1}^k}{\partial x_1} & \cdots & \frac{\partial \psi_{r,1}^k}{\partial x_d} & \frac{\partial \psi_{r,1}^k}{\partial y_1} & \cdots & \frac{\partial \psi_{r,1}^k}{\partial y_d} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \psi_{r,d}^k}{\partial x_1} & \cdots & \frac{\partial \psi_{r,d}^k}{\partial x_d} & \frac{\partial \psi_{r,d}^k}{\partial y_1} & \cdots & \frac{\partial \psi_{r,d}^k}{\partial y_d} \end{pmatrix}, \quad \|J_{\psi_r^k}(\boldsymbol{x}, \boldsymbol{y})\| \leq R, \tag{17}$$

$$J_{\psi_a^k}(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial \psi_{a,1}^k}{\partial x_1} & \cdots & \frac{\partial \psi_{a,1}^k}{\partial x_d}, \\ \vdots & \ddots & \vdots \\ \frac{\partial \psi_{a,d}^k}{\partial x_1} & \cdots & \frac{\partial \psi_{a,d}^k}{\partial x_d} \end{pmatrix}, \quad \|J_{\psi_a^k}(\boldsymbol{x})\| \leq A. \tag{18}$$

Then, given the initial state $(t_0, \boldsymbol{z}_1^{t_0}, \cdots, \boldsymbol{z}_N^{t_0}, \boldsymbol{w}_1, \cdots, \boldsymbol{w}_N)$, we claim that there exists $\varepsilon > 0$, such that the ODE system Eqn. 10 has a unique solution in the interval $[t_0 - \varepsilon, t_0 + \varepsilon]$.

We first introduce the Picard–Lindelöf Theorem as below.

Theorem A.1. *(Picard–Lindelöf Theorem) Let $D \subseteq \mathbb{R} \times \mathbb{R}^n$ be a closed rectangle with $(t_0, y_0) \in D$. Let $f : D \to \mathbb{R}^n$ be a function that is continuous in t and Lipschitz continuous in y. Then, there exists some $\varepsilon > 0$ such that the initial value problem:*

$$y'(t) = f(t, y(t)), \quad y(t_0) = y_0. \tag{19}$$

has a unique solution $y(t)$ on the interval $[t_0 - \varepsilon, t_0 + \varepsilon]$.

Then, we prove the following lemma.

Lemma A.1. *Suppose we have a series of L-Lipschitz continuous functions $\{f_i : \mathbb{R}^m \to \mathbb{R}^n\}_{i=1}^N$, and then their linear combination is also L-Lipschitz continuous, i.e., $\forall \{a_1, \cdots a_N\} \in [0, 1]^N$, satisfying $\sum_{i=1}^N a_i = 1$, we have $\sum_{i=1}^N a_i f_i$ is also L-Lipschitz continuous.*

Proof. $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^m$, we have:

$$\begin{aligned} \|\sum_{i=1}^N a_i f_i(\boldsymbol{x}) - \sum_{i=1}^N a_i f_i(\boldsymbol{y})\| &\leq \sum_{i=1}^N a_i \|f_i(\boldsymbol{x}) - f_i(\boldsymbol{y})\| && (20) \\ &\leq \sum_{i=1}^N a_i L \|\boldsymbol{x} - \boldsymbol{y}\| && (21) \\ &= L\|\boldsymbol{x} - \boldsymbol{y}\|. && (22) \end{aligned}$$

□

Next, we show the proof of Lemma 3.1.

Proof. First, we can rewrite the ODE system Eqn. 10 as:

$$\frac{d\boldsymbol{Z}^t}{dt} = \sum_{k=1}^K \boldsymbol{W}^k f^k(\boldsymbol{Z}^t) - \boldsymbol{Z}^t, \tag{23}$$

where $\boldsymbol{W}^k \in \mathbb{R}^{Nd \times Nd}$ is a diagonal matrix. It is evident that the right hand side is continuous with respect to t since it does not depend on t directly.

Then, for any continuous function $f : \mathbb{R}^n \to \mathbb{R}^m$, with the Mean Value Theorem, we have $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \|f(\boldsymbol{x}) - f(\boldsymbol{y})\| = \|J_f(\boldsymbol{p})\| * \|\boldsymbol{x} - \boldsymbol{y}\|$, where $\boldsymbol{p}$ is a point in the segment connecting $\boldsymbol{x}$ and $\boldsymbol{y}$.

Now, denote $A(i,j) \in \mathbb{R}^{2\times dN}$ with the first row has elements with index from idN+1 to (i+1)dN be 1, the others 0; the second row has elements with index from jdN+1 to (j+1)dN be 1, the others 0. By introducing $A(i,j)$, for all $\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_N \end{pmatrix}, \boldsymbol{Y} = \begin{pmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_N \end{pmatrix} \in \mathbb{R}^{dN}$, we have:

$$\|\psi_r^k(A(i,j)\boldsymbol{X}) - \psi_r^k(A(i,j)\boldsymbol{Y})\| \leq \|\psi_r^k(\boldsymbol{x}_i,\boldsymbol{x}_j) - \psi_r^k(\boldsymbol{y}_i,\boldsymbol{x}_j)\| + \|\psi_r^k(\boldsymbol{y}_i,\boldsymbol{x}_j) - \psi_r^k(\boldsymbol{y}_i,\boldsymbol{y}_j)\| \tag{24}$$

$$= \|J_{\psi_r^k}(\boldsymbol{p}_i)\| * \|\boldsymbol{x}_i - \boldsymbol{y}_i\| + \|J_{\psi_r^k}(\boldsymbol{p}_j)\| * \|\boldsymbol{x}_j - \boldsymbol{y}_j\| \tag{25}$$

$$\leq R\|\boldsymbol{x}_i - \boldsymbol{y}_i\| + R\|\boldsymbol{x}_j - \boldsymbol{y}_j\| \tag{26}$$

$$\leq R\|\boldsymbol{X} - \boldsymbol{Y}\|, \tag{27}$$

where $\boldsymbol{p}_i$ is a point in the segment connecting $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, and a similar definition is for $\boldsymbol{p}_j$. Note that we have ψ_r^k is R-Lipschitz continuous. Therefore, by Lemma A.1, the following linear combination is also R-Lipschitz continuous:

$$l^k(\boldsymbol{Z}^t) = \sum_{j^t \in \mathcal{S}(i^t)} \psi_r^k([A(i^t,j^t)\boldsymbol{Z}^t]). \tag{28}$$

Thus, for all $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{dN}$, we have:

$$\|f^k(\boldsymbol{X}) - f^k(\boldsymbol{Y})\| = \|\psi_a^k(l^k(\boldsymbol{X})) - \psi_a^k(l^k(\boldsymbol{Y}))\| \tag{29}$$

$$\leq A\|l^k(\boldsymbol{X}) - l^k(\boldsymbol{Y})\| \tag{30}$$

$$\leq AR\|\boldsymbol{X} - \boldsymbol{Y}\|. \tag{31}$$

Again, we have each f^k is AR-Lipschitz continuous, so their linear combination $\sum_{k=1}^K \boldsymbol{W}^k f^k$ will also be AR-Lipschitz continuous. Finally, we have

$$\|[\sum_{k=1}^K \boldsymbol{W}^k f^k(\boldsymbol{X}) - \boldsymbol{X}] - [\sum_{k=1}^K \boldsymbol{W}^k f^k(\boldsymbol{Y}) - \boldsymbol{Y}]\| \leq \|\sum_{k=1}^K \boldsymbol{W}^k f^k(\boldsymbol{X}) - \sum_{k=1}^K \boldsymbol{W}^k f^k(\boldsymbol{Y})\| \tag{32}$$

$$+ \|\boldsymbol{X} - \boldsymbol{Y}\| \tag{33}$$

$$\leq (AR+1)\|\boldsymbol{X} - \boldsymbol{Y}\|. \tag{34}$$

Thus, the right hand side will be (AR+1)-Lipschitz continuous. According to the Theorem A.1, we prove the uniqueness of the solution to Eqn. 10. □

B Related Work

B.1 Interacting Dynamics Modeling

Recent years have witnessed a surge of interest in modeling interacting dynamical systems across a variety of fields including molecular biology and computational physics (Shao et al., 2022; Lan et al., 2022; Li et al., 2022b; Bishnoi et al., 2022). While convolutional neural networks (CNNs) have been successfully employed to learn from regular data such as grids and frames (Peng et al., 2020), emerging research is increasingly utilizing geometric graphs to represent more complex systems (Wu et al., 2023; Deng et al., 2023). Graph neural networks (GNNs) have thus become increasingly prevailing for modeling these intricate dynamics (Pfaff et al., 2021; Shao et al., 2022; Sanchez-Gonzalez et al., 2020; Allen et al., 2022; Look et al., 2023; Yıldız et al., 2022). AgentFormer (Yuan et al., 2021) jointly models both time and social dimensions with semantic information preserved. NRI (Kipf et al., 2018) models interactions along with node states from observations using GNNs. R-SSM (Yang et al., 2020) models the dynamics of interacting objects using GNNs and includes auxiliary contrastive prediction tasks to enhance discriminative learning. Despite their popularity, current methods often fall short in modeling challenging scenarios such as out-of-distribution shift and long-term dynamics (Yu et al., 2021). To address these limitations, our work leverages contextual knowledge to incorporate prototype decomposition into a graph ODE framework.

B.2 Neural Ordinary Differential Equations

Motivated by the approximation of residual networks (Chen et al., 2018), neural ordinary differential equations (ODEs) have been introduced to model continuous-time dynamics using parameterized derivatives in hidden spaces. These neural ODEs have found widespread use in time-series forecasting due to their effectiveness (Dupont et al., 2019; Xia et al., 2021; Jin et al., 2022; Schirmer et al., 2022). Incorporated with the message passing mechanism, they have been integrated with GNNs, which can mitigate the issue of oversmoothing and enhance model interpretability (Xhonneux et al., 2020; Zhang et al., 2022; Poli et al., 2019). I-GPODE (Yıldız et al., 2022) estimates the uncertainty of trajectory predictions using the Gaussian process, which facilitates effective long-term predictions. HOPE (Luo et al., 2023) focuses on incorporating second-order graph ODE in evolution modeling. In contrast, our method introduces hierarchical context discovery with disentanglement to guide the prototype decomposition of individual nodes in modeling interacting dynamics.

B.3 Graph Neural Networks

Graph Neural Networks (GNNs) (Kipf & Welling, 2017; Xu et al., 2019a; Veličković et al., 2018) have shown remarkable efficacy in handling a range of graph-based machine learning tasks such as node classification (Yang et al., 2021) and graph classification (Liu et al., 2022). Typically, they adopt the message passing mechanism, where each node aggregates messages from its adjacent nodes for updated node representations. Recently, researchers have started to focus more on realistic graphs that do not obey the homophily assumption and developed several GNN approaches to tackle heterophily (Zhu et al., 2021; Li et al., 2022a; Zhu et al., 2020). These approaches typically leverage new graph structures (Zhu et al., 2020; Suresh et al., 2021) and modify the message passing procedures (Chien et al., 2021; Yan et al., 2022) to mitigate the influence of potential heterophily. In our PGODE, we focus on interacting dynamics systems instead. In particular, due to the local heterophily, different objects should have different interacting patterns, and therefore we infer object-level contexts from historical data.

C More Discussion About Expressivity

We provide more discussion about the expressivity of the proposed PGODE. Piecewise continuous neural networks have been proven asymptotically more expressive than classical feed forward networks (Kratsios & Zamanlooy, 2022). Our prototype decomposition adopts a soft form of piecewise functions to enhance the expressivity, which can also help capture the influence of seasonality and events in real-world dynamics systems. Our empirical results in ID settings also validate the strong expressivity when handling complicated dynamics.

D Algorithm

We summarize the learning algorithm of our PGODE in Algorithm 1.

E Detail of Baselines

Our approach is compared with various baselines for dynamics systems modeling, i.e., LSTM (Hochreiter & Schmidhuber, 1997), GRU (Weerakody et al., 2021), NODE (Chen et al., 2018), LG-ODE (Huang et al., 2020), MPNODE (Chen et al., 2022), SocialODE (Wen et al., 2022) and HOPE (Luo et al., 2023).

The proposed method is compared with seven competing baselines as follows:

- LSTM (Hochreiter & Schmidhuber, 1997) has been broadly utilized for sequence prediction tasks. Compared with classic RNNs, LSTM incorporates three critical gates, i.e., the forget gate, the input gate, and the output gate, which can effectively understand and retain important long-term dependencies within the data sequences.

Algorithm 1 Training Algorithm of PGODE

Input: The observations $G^{1:T} = \{G^1, \cdots, G^T\}$.
Output: The parameters in our model.

```
Initialize model parameters;
while not convergence do
  for each training sequence do
    Partition the sequence into two segments;
    Construct the temporal graph using Eqn. 2;
    Generate object-level contexts using Eqn. 5;
    Generate system-level contexts with summarization;
    Solve our prototypical graph ODE in Eqn. 10;
    Output the trajectories using the decoder;
    Calculate the final loss in Eqn. 16;
    Update τ' in our PGODE using gradient ascent;
    Update other parameters in our PGODE using gradient descent;
  end for
end while
```

- GRU (Cho et al., 2014) is another popular RNN architecture, which employs the gating mechanism to control the information flow during propagation. GRU has an improved computational efficiency compared LSTM.
- NODE (Chen et al., 2018) is the first method to introduce a continuous neural network based on the residual connection. It has been shown effective in time-series forecasting.
- LG-ODE (Huang et al., 2020) incorporates graph neural networks with neural ODE, which can capture continuous interacting dynamics in irregularly-sampled partial observations.
- MP-NODE (Gupta et al., 2022) combines the message passing mechanism and neural ODEs, which can capture sub-system relationships during the evolution of homogeneous systems.
- SocialODE (Wen et al., 2022) simulates the evolution of agent states and their interactions using a neural ODE architecture, which shows remarkable performance in multi-agent trajectory forecasting.
- HOPE (Luo et al., 2023) is a recently proposed graph ODE method, which adopts a twin encoder to learn latent state representations. These representations are fed into a high-order graph ODE to learn long-term correlations from complicated dynamical systems.

F Dataset Details

We use four simulation datasets to evaluate our proposed GOAT, including physical and molecular dynamic systems. We will introduce the details of these four datasets in this part.

- *Springs* & *Charged*. The two physical dynamic simulation datasets *Springs* and *Charged* are commonly used in the field of machine learning for simulating physical systems. The *Springs* dataset simulates a system of interconnected springs governed by Hooke's law. Each spring has inherent properties such as elasticity coefficients and initial positions, representing a dynamic mechanical system. Each sample in the *Springs* dataset contains 10 interacting springs with information about the current state, i.e., velocity and acceleration, and additional properties, i.e., mass and damping coefficients. Similar to the *Springs* dataset, *Charged* is another popular physical dynamic simulation dataset that simulates electromagnetic phenomena. The objects in *Charged* are replaced by the electronics. We use the box size α, the initial velocity β, the interaction strength γ, and springcharged probability δ as the system parameters in the experiments. It is noteworthy that the objects attract or repel with equal probability in the *Charged* system but unequal probability in the spring system. Both systems have a given graph indicating fixed interactions from real springs or electric charge effects.

- *5AWL & 2N5C*. To evaluate our approach on modeling molecular dynamic systems, we construct two datasets from two proteins, *5AWL* and *2N5C*, which can be accessed from the RCSB[1]. First, we repair missing residues, non-standard residues, missing atoms, and hydrogen atoms in the selected protein. Additionally, we adjust the size of the periodic boundary box to ensure that it is sufficiently large, thus avoiding truncation effects and abnormal behavior of the simulation system during the data simulation process. Then, we perform simulations on the irregular molecular motions within the protein using Langevin Dynamics (García-Palacios & Lázaro, 1998) under the NPT (isothermal-isobaric ensemble) conditions, with parameters sampled from the specified range, and we extract a frame every 0.2 ps to record the protein structure, which constitutes the dataset used for supervised learning. In the two constructed datasets, we use the temperature t, pressure value p, and frictional coefficient μ as the dynamic system parameters. Langevin Dynamics is a mathematical model used to simulate the flow dynamics of molecular systems (Bussi & Parrinello, 2007). It can simplify complex systems by replacing some degrees of freedom of the molecules with stochastic differential equations. For a dynamic system containing N particles of mass m, with coordinates given by $X = X(t)$, the Langevin equation of it can be formulated as follows:

$$m\frac{d^2X}{dt^2} = -\Delta U(X) - \mu\frac{dX}{dt} + \sqrt{2\mu k_b T}R(t), \tag{35}$$

where μ represents the frictional coefficient, $\Delta U(X)$ is the interaction potential between particles, Δ is the gradient operator, T is the temperature, k_b is Boltzmann constant and $R(t)$ is delta-correlated stationary Gaussian process.

G Implementation Details

Table 4: Datasets and distributions of system parameters. For the OOD test set, there is at least one of the system parameters outside the range utilized for training. α: box size, β: initial velocity norm,γ: interaction strength, δ: spring/charged probability. t: temperature, p: pressure, μ: frictional coefficient.

	Springs	*Charged*	*5AWL/2N5C*
Parameters	$\alpha, \beta, \gamma, \delta$	$\alpha, \beta, \gamma, \delta$	t, p, μ
Train/Val/Test	$A = \{\alpha \in [4.9, 5.1]\}$ $B = \{\beta \in [0.49, 0.51]\}$ $C = \{\gamma \in [0.09, 0.11]\}$ $D = \{\delta \in [0.49, 0.51]\}$ $\Omega_{\text{train}} = (A \times B \times C \times D)$	$A = \{\alpha \in [4.9, 5.1]\}$ $B = \{\beta \in [0.49, 0.51]\}$ $C = \{\gamma \in [0.9, 1.1]\}$ $D = \{\delta \in [0.49, 0.51]\}$ $\Omega_{\text{train}} = (A \times B \times C \times D)$	$T = \{t \in [290, 310]\}$ $P = \{p \in [0.9, 1.1]\}$ $M = \{\mu \in [0.9, 1.1]\}$ $\Omega_{\text{train}} = (T \times P \times M)$
OOD Test Set	$A = \{\alpha \in [4.8, 5.2]\}$ $B = \{\beta \in [0.48, 0.52]\}$ $C = \{\gamma \in [0.08, 0.12]\}$ $D = \{\delta \in [0.48, 0.52]\}$ $\Omega_{\text{OOD}} =$ $(A \times B \times C \times D) \setminus \Omega_{\text{train}}$	$A = \{\alpha \in [4.8, 5.2]\}$ $B = \{\beta \in [0.48, 0.52]\}$ $C = \{\gamma \in [0.8, 1.2]\}$ $D = \{\delta \in [0.48, 0.52]\}$ $\Omega_{\text{OOD}} =$ $(A \times B \times C \times D) \setminus \Omega_{\text{train}}$	$T = \{t \in [280, 320]\}$ $P = \{p \in [0.8, 1.2]\}$ $M = \{\mu \in [0.8, 1.2]\}$ $\Omega_{\text{OOD}} =$ $(T \times P \times M) \setminus \Omega_{\text{train}}$
Number of samples			
Train/Val/Test	1000/200/200		200/50/50
OOD Test Set	200		50

In our experiments, we employ a rigorous data split strategy to ensure the accuracy of our results. Specifically, we split the whole datasets into four different parts, including the normal three sets, i.e., training, validating and in-distribution (ID) test sets and an out-of-distribution (OOD) test set. For the physical dynamic datasets, we generate 1200 samples for training and validating, 200 samples for ID testing and 200 samples for OOD testing. For the molecular dynamic datasets, we construct 200 samples for training, 50 samples for validating, 50 samples for ID testing and 50 samples for testing in OOD settings.

Each sample in the datasets has a group of distinct system parameters as shown in Table 4. For training, validation and ID test samples, we randomly sample system parameters in the space of

[1]https://www.rcsb.org

Ω_{train}. For OOD samples, the system parameters come from Ω_{OOD} randomly, which indicates distribution shift compared with the training domain. During the training process, each trajectory sample is further split into two parts, i.e., a conditional part for initializing object-level contexts representation and global-level contexts representation, and a prediction part for supervising the model. The size of the two parts is denoted as conditional length and prediction length, respectively. In our experiments, we set the conditional length to 12, and we used three different prediction lengths, i.e., 12, 24, and 36.

We adopt PyTorch (Paszke et al., 2017) and torchdiffeq package (Kidger et al., 2021) to implement all the compared approaches and our PGODE. All these experiments in this work are performed on a single NVIDIA A40 GPU. The fourth-order Runge-Kutta method from torchdiffeq is adopted as the ODE solver. We employ a set of one-layer GNN prototypes with a hidden dimension of 128 for graph ODE. The number of prototypes is set to 5 as default. For optimization, we utilize an Adam optimizer (Kingma & Ba, 2015) with an initial learning rate of 0.0005. The batch size is set to 256 for the physical dynamic simulation datasets and 64 for the molecular dynamic simulation datasets.

H More Experiment Results

H.1 Performance Comparison

To begin, we compare with our PGODE with more baselines, i.e., AgentFormer (Yuan et al., 2021), NRI (Kipf et al., 2018) and I-GPODE (Yıldız et al., 2022) in our performance comparison. The results of these comparisons are presented in Table 5 and our method outperforms the compared methods. In addition, we show the performance of the compared methods in two different coordinates of positions and velocities, i.e., q_x, q_y, v_x and v_y. The compared results on *Springs* and *Charged* are shown in Table 6 and Table 7, respectively. From the results, we can observe the superiority of the proposed PGODE in capturing complicated interacting patterns under both ID and OOD settings.

Table 5: Performance comparison with NRI, AgentFormer, and I-GPODE on physical dynamics simulations (MSE $\times 10^{-2}$). NRI, AgentFormer, and I-GPODE are out of memory on molecular dynamics simulations.

Prediction Length	12 (ID)		24 (ID)		36 (ID)		12 (OOD)		24 (OOD)		36 (OOD)	
Variable	q	v	q	v	q	v	q	v	q	v	q	v
Springs												
NRI	0.103	0.425	0.210	0.681	0.693	2.263	0.119	0.472	0.246	0.770	0.807	2.406
AgentFormer	0.115	0.163	0.202	0.517	1.656	1.691	0.157	0.195	0.243	0.505	1.875	1.913
I-GPODE	0.159	0.479	0.746	3.002	1.701	7.433	0.173	0.498	0.796	3.193	1.818	7.322
PGODE (Ours)	**0.035**	**0.124**	**0.070**	**0.262**	**0.296**	**1.326**	**0.047**	**0.138**	**0.088**	**0.291**	**0.309**	**1.337**
Charged												
NRI	0.901	2.702	3.225	3.346	7.770	4.543	1.303	2.726	3.678	3.548	8.055	4.752
AgentFormer	1.076	2.476	3.631	3.044	7.513	3.944	1.384	2.514	4.224	3.199	8.985	4.002
I-GPODE	1.044	2.818	3.407	3.751	7.292	4.570	1.322	2.715	3.805	3.521	8.011	4.056
PGODE (Ours)	**0.578**	**2.196**	**2.037**	**2.648**	**4.804**	**3.551**	**0.802**	**2.135**	**2.584**	**2.663**	**5.703**	**3.703**

H.2 Ablation Study

We show more ablation studies on *Charged* and *2N5C* to make our analysis complete. In particular, the compared performance of different model variants are shown in Table 8. From the results, we can observe that our full model can outperform all the model variance in all cases, which validates the effectiveness of each component in our PGODE again. In addition, we introduce two model variants: (1) PGODE w. MLP, which combines a GNN with an MLP to learn the individualized dynamics; (2) PGODE w. Single, which takes the node representation and the global representation as input with a single message passing function. The compared performance of different model variants is shown in Table 9. From the results, we can observe that our full model can outperform all the model variance in all cases. Compared with these variants, our prototype decomposition can involve different GNN bases, which model diverse evolving patterns to jointly determine the individualized dynamics. This strategy can enhance the model expressivity, allowing for more accurate representation learning of hierarchical structures from a mixture-of-experts perspective.

Table 6: Mean Squared Error (MSE) $\times 10^{-2}$ on *Springs*.

Prediction Length	12				24				36			
Variable	q_x	q_y	v_x	v_y	q_x	q_y	v_x	v_y	q_x	q_y	v_x	v_y
ID												
LSTM	0.324	0.250	0.909	0.931	0.679	0.638	2.695	2.623	1.253	1.304	5.023	6.434
GRU	0.496	0.291	0.565	0.628	0.873	0.623	1.711	2.001	1.368	1.128	2.980	3.912
NODE	0.165	0.148	0.649	0.479	0.722	0.621	2.534	2.293	1.683	1.534	6.323	6.142
LG-ODE	0.077	0.077	0.264	0.272	0.174	0.135	0.449	0.576	0.613	0.441	1.757	2.528
MPNODE	0.080	0.072	0.222	0.263	0.237	0.105	0.407	0.506	0.866	0.335	1.469	2.006
SocialODE	0.069	0.068	0.205	0.315	0.138	0.120	0.391	0.630	0.429	0.400	1.751	2.624
HOPE	0.087	0.053	0.152	0.200	0.571	0.342	0.707	1.206	2.775	2.175	4.412	6.405
PGODE (Ours)	**0.033**	**0.037**	**0.122**	**0.127**	**0.074**	**0.066**	**0.239**	**0.286**	**0.318**	**0.273**	**1.186**	**1.466**
OOD												
LSTM	0.499	0.449	1.086	1.227	1.019	0.857	2.847	2.466	1.768	1.415	5.154	5.293
GRU	0.714	0.469	0.713	0.703	1.280	0.905	1.795	2.096	1.844	1.497	2.852	3.994
NODE	0.246	0.209	0.997	0.585	0.876	0.687	2.790	2.269	2.002	1.663	6.349	5.670
LG-ODE	0.093	0.083	0.272	0.327	0.185	0.172	0.463	0.661	0.684	0.545	1.767	2.645
MPNODE	0.107	0.081	0.230	0.268	0.299	0.126	0.420	0.528	0.967	0.386	1.464	1.969
SocialODE	0.082	0.076	0.221	0.350	0.151	0.156	0.414	0.726	0.488	0.495	1.793	2.826
HOPE	0.094	0.058	0.178	0.264	0.506	0.523	1.031	1.603	2.369	2.251	3.701	8.291
PGODE (Ours)	**0.046**	**0.048**	**0.133**	**0.144**	**0.094**	**0.081**	**0.286**	**0.297**	**0.336**	**0.281**	**1.360**	**1.313**

Table 7: Mean Squared Error (MSE) $\times 10^{-2}$ on *Charged*.

Prediction Length	12				24				36			
Variable	q_x	q_y	v_x	v_y	q_x	q_y	v_x	v_y	q_x	q_y	v_x	v_y
ID												
LSTM	0.743	0.846	2.913	3.145	2.797	3.052	3.605	3.863	6.477	6.660	4.240	4.423
GRU	0.764	0.799	2.931	3.063	2.709	2.901	3.572	3.709	5.657	6.281	4.068	4.227
NODE	0.743	0.808	2.764	2.777	2.913	3.114	3.432	3.451	6.468	6.868	3.997	4.089
LG-ODE	0.736	0.783	2.322	2.414	2.320	2.731	3.361	3.268	5.188	6.782	6.194	5.043
MPNODE	0.720	0.759	2.414	2.496	2.379	2.536	3.589	3.738	5.636	5.614	5.472	7.046
SocialODE	0.630	0.695	2.311	2.358	2.252	2.631	3.509	2.995	5.743	7.076	5.701	4.122
HOPE	0.593	0.635	2.295	2.337	3.214	2.938	3.279	3.482	9.289	7.845	8.406	8.511
PGODE (Ours)	**0.555**	**0.600**	**2.164**	**2.228**	**1.940**	**2.134**	**2.624**	**2.673**	**4.449**	**5.159**	**3.778**	**3.324**
OOD												
LSTM	1.130	1.123	3.062	2.992	4.026	3.950	3.768	3.512	7.934	8.435	4.517	3.925
GRU	1.072	1.012	3.108	2.948	3.893	3.602	3.844	3.428	6.970	8.061	4.485	3.718
NODE	1.185	1.062	2.956	2.732	4.057	3.804	3.645	3.480	8.622	8.372	5.097	4.376
LG-ODE	0.999	0.866	2.581	2.521	2.797	3.239	4.200	2.978	5.996	7.593	8.422	4.309
MPNODE	1.092	0.897	2.487	2.623	2.967	2.828	3.670	4.001	6.051	6.118	6.029	7.566
SocialODE	0.865	0.924	2.481	2.359	2.610	3.177	3.968	2.836	**5.482**	7.102	8.530	4.150
HOPE	0.839	0.918	2.466	2.484	3.586	3.783	3.417	3.442	11.254	10.652	10.133	8.107
PGODE (Ours)	**0.739**	**0.865**	**2.159**	**2.110**	**2.524**	**2.643**	**2.704**	**2.623**	5.748	**5.659**	**4.017**	**3.389**

Table 8: Ablation study on *Charged* (MSE $\times 10^{-2}$) and *2N5C* (MSE $\times 10^{-3}$) with a prediction length of 24.

Dataset	*Charged* (ID)		*Charged* (OOD)		*2N5C* (ID)			*2N5C* (OOD)		
Variable	q	v	q	v	q_x	q_y	q_z	q_x	q_y	q_z
PGODE w/o O	2.282	3.013	2.590	2.943	2.076	2.130	2.215	2.582	2.800	2.833
PGODE w/o S	2.308	2.994	2.990	2.911	2.040	2.046	2.227	2.559	2.791	2.854
PGODE w/o F	2.497	3.298	2.882	3.197	2.424	2.208	2.465	2.970	2.868	3.118
PGODE w/o D	2.179	2.842	2.616	3.076	2.119	2.083	2.171	2.785	2.759	2.829
PGODE (Full Model)	**2.037**	**2.648**	**2.584**	**2.663**	**1.960**	**2.029**	**2.119**	**2.464**	**2.734**	**2.727**

Table 9: Further ablation study on *Springs* (MSE $\times 10^{-2}$) and *5AWL* (MSE $\times 10^{-3}$) with a prediction length of 24.

Dataset	*Springs* (ID)		*Springs* (OOD)		*5AWL* (ID)			*5AWL* (OOD)		
Variable	q	v	q	v	q_x	q_y	q_z	q_x	q_y	q_z
PGODE w. Single	0.208	0.434	0.248	0.481	3.010	3.741	3.143	3.523	4.691	3.839
PGODE w. MLP	0.152	0.454	0.179	0.514	2.997	3.638	3.240	3.605	4.492	3.908
PGODE (Full Model)	**0.070**	**0.262**	**0.088**	**0.291**	**2.910**	**3.384**	**2.904**	**3.374**	**4.334**	**3.615**

H.3 Performance with Different Number of Prototypes

Figure 5 (a) (b) (c) and (d) record the performance with respect to different numbers of prototypes on different datasets. From the results, we can find that more prototypes would bring in better results before saturation.

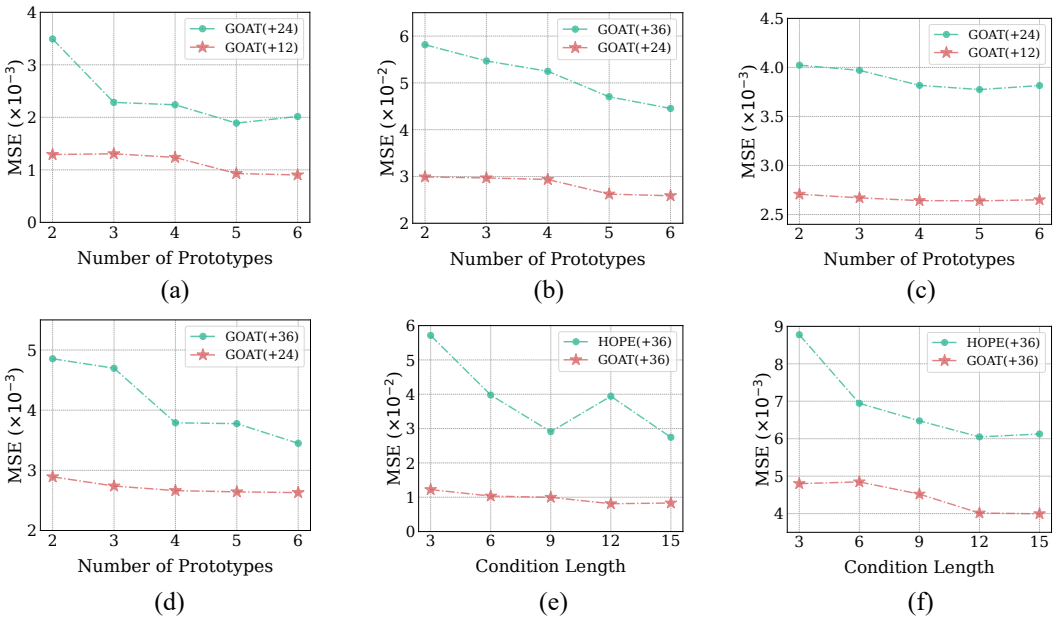

Figure 5: (a),(b),(c),(d) Performance on the OOD test set of *Springs*, *Charged*, *5AWL*, and *2N5C* with respect to four different numbers of prototypes. (e),(f) Performance with respect to different condition lengths on the ID test set of *Springs* and *5AWL*.

H.4 Performance with Different Condition Lengths

We analyze the influence of different conditional lengths by varying them in $\{3, 6, 9, 12, 15\}$, respectively. As shown in Figure 5 (e) and (f), we can observe that our PGODE can always outperform the latest baseline HOPE, which validates the superiority of the proposed PGODE.

H.5 Efficiency Comparison

We have conducted a comparison of computation cost. The results are shown in Table 10 and we can observe that our method has a competitive computation cost. In particular, the performance of HOPE is much worse than ours (the increasement of ours is over 47% compared with HOPE), while our computational burden only increases a little. Moreover, both the performance and efficiency of I-GPODE are worse than ours.

Table 10: Comparison of training cost per epoch (s).

Method	LSTM	GRU	NODE	LG-ODE	MPNODE	SocialODE	I-GPODE	HOPE	PGODE (Ours)
Springs	1.53	1.04	2.21	17.39	23.33	21.02	267.08	23.86	37.03
Charged	1.33	1.02	2.06	16.59	22.26	19.93	250.23	20.43	33.88

H.6 VISUALIZATION

In addition, we present more visualization of the proposed PGODE and two baselines, i.e., SocialODE and HOPE. We have offered visualization of the predicted trajectory of a sample in Figure 2 and now we visualize four extra test instances (two ID samples and two OOD samples) in Figure 6. From the results, we can observe that the proposed PGODE is capable of generating more reliable trajectories in comparison to the baselines. For instance, our PGODE can discover the correct direction of the orange particle while the others fail in the second OOD instance.

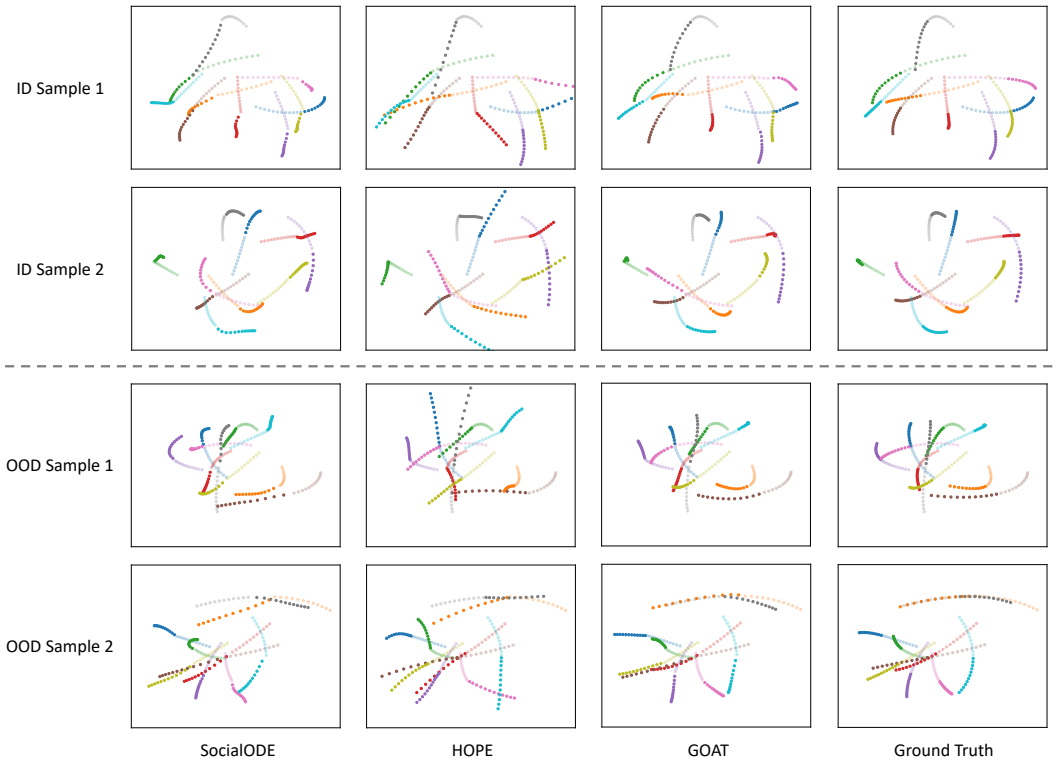

Figure 6: Visualization of different methods on *Springs*. Semi-transparent paths denote observed trajectories, from which the latent initial states are estimated. Solid paths denote model predictions.